Project Documentation: Trade Fraud Detector

Project Overview

The Trade Fraud Detector project is designed to identify anomalies and potentially fraudulent transactions within trade datasets. It combines simple rule-based techniques with machine learning methods to offer a robust yet lightweight detection system. The primary goal is to assist organizations in identifying suspicious trade behavior early, using a dataset containing transaction details like invoice IDs, buyer/seller information, transaction amounts, and product details.

This system is ideal for internal use by compliance teams or analysts and can be customized to suit different trade environments. With the addition of a user-friendly Streamlit interface, the tool allows non-technical users to interact with the detection logic by simply uploading a CSV file.

Project Structure

The project has been modularized into distinct components to separate the detection logic, user interface, and data handling. Here is a breakdown of each part:

- **app.py**: The entry point for the Streamlit dashboard. It allows users to upload a CSV file, view flagged anomalies, and see data visualizations.
- **src/trade_fraud_detector.py**: This module contains all core logic for fraud detection. It includes preprocessing steps, rule-based anomaly detection, machine learning-based detection using Isolation Forest, and functions to generate relevant plots.
- **src/generate_trade_data.py**: Script for generating synthetic trade data. Useful when actual data is unavailable or restricted.
- **data/**: Stores the uploaded input datasets and output anomaly reports.
- **output/**: Contains generated plots like outlier scatterplots and invoice trends.
- **requirements.txt**: Lists all Python dependencies required to run the project.

This modular approach enhances readability, maintainability, and flexibility. Each module can be updated or replaced independently, making the system easy to evolve based on user or business requirements.

Purpose of the Dataset Generator

In many real-world cases, acquiring clean and labeled datasets can be a challenge due to privacy concerns or lack of data access. The included dataset generator script addresses this issue by simulating realistic trade transaction data.

The generator allows users to:

- Create large volumes of structured data instantly
- Introduce controlled anomalies such as duplicate invoice IDs and unusually high transaction amounts
- Test the detection pipeline in the absence of real data
- Demonstrate the system to stakeholders using synthetic but realistic datasets

This makes the solution testable and demonstrable in isolated environments, allowing for iterative improvements before integrating with actual business data.

Project Strengths

- Combines rule-based and machine learning approaches for better anomaly coverage
- Easy-to-use web interface using Streamlit
- Modular architecture suitable for integration and scaling
- Visual feedback via plots to support anomaly interpretation
- Custom dataset generator to facilitate testing and demos
- Can be extended with more sophisticated models or integrated into production pipelines

How to Implement and Run the Project

1. Set up the Environment
    - •Clone the repository to your local system
    - • install dependencies using pip install -r requirements.txt

2. Generate a Dataset (Optional)
    - Run python src/generate_trade_data.py
3. Launch the Streamlit App
    - From the project root, run streamlit run app.py
    - Use the interface to upload a CSV file and review flagged anomalies
4. View Results
    - Flagged transactions will be displayed in a table
    - Output files like plots and reports will be saved in the output/ and data/ directories

Conclusion

The Trade Fraud Detector project provides a practical and adaptable approach to fraud detection in trade datasets. With its modular design, rule-ML hybrid detection strategy, and interactive frontend, it serves as a strong foundation for further expansion. Whether you're presenting to stakeholders, testing hypotheses, or seeking an early-warning system for transactional anomalies, this tool is structured to be effective and extensible in real-world applications.